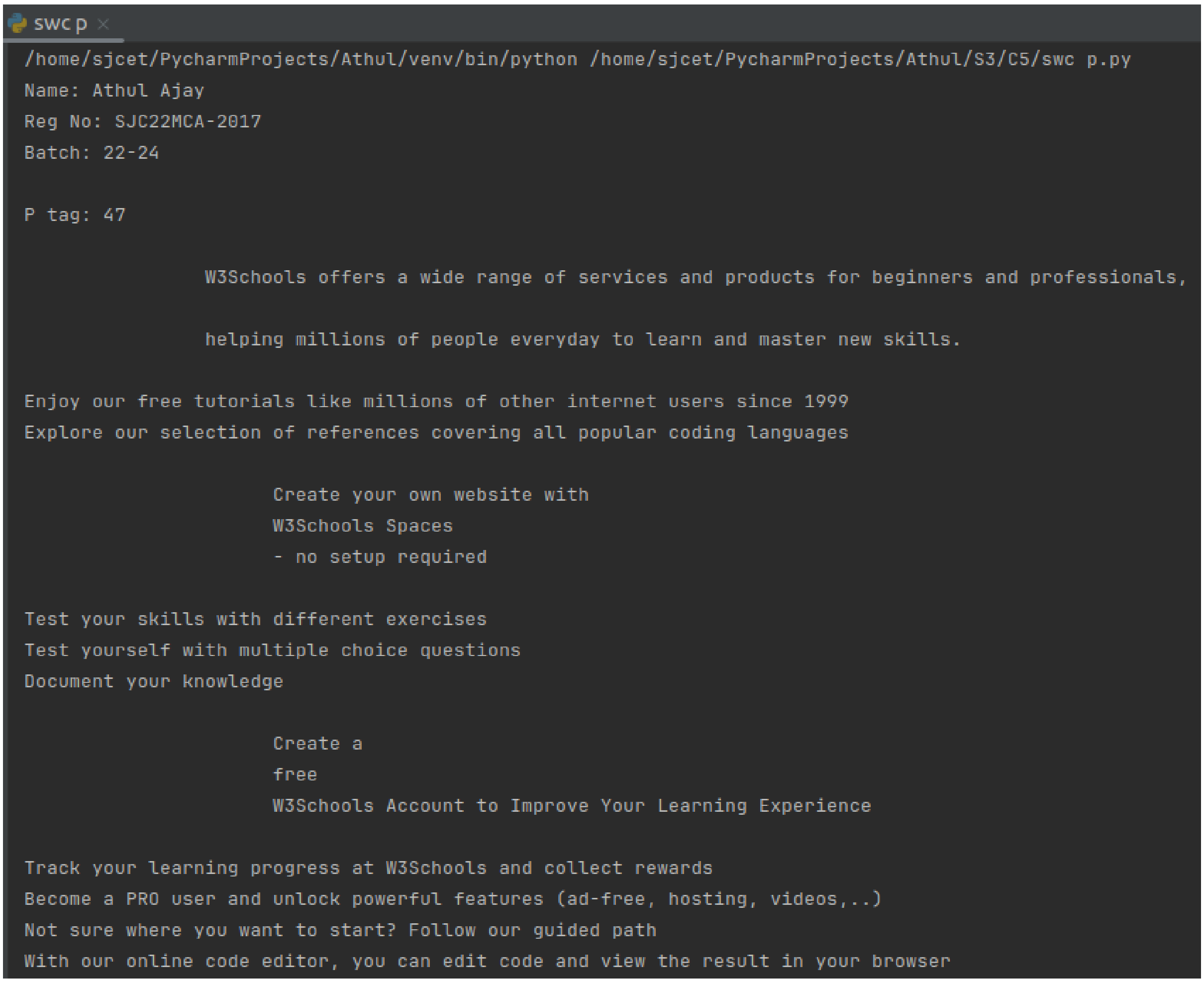
Lab Cycle 5

1.Write a program to implement simple web crawler using Python extract and display the content of the page (p tag)

Input code:

```
import requests
from bs4 import BeautifulSoup

def getdata(url):
    r = requests.get(url)
    return r.content

htmldata = getdata("https://www.w3schools.com/python/python_ml_scale.asp")
soup = BeautifulSoup(htmldata, 'html.parser')
data = ''
print("Name: Athul Ajay")
print("Reg No: SJC22MCA-2017")
print("Batch: 22-24")
print()
pr = len(soup.find_all('p'))
print("P tag:", pr)

for data in soup.find_all('p'):
    print(data.get_text())
```

Output:



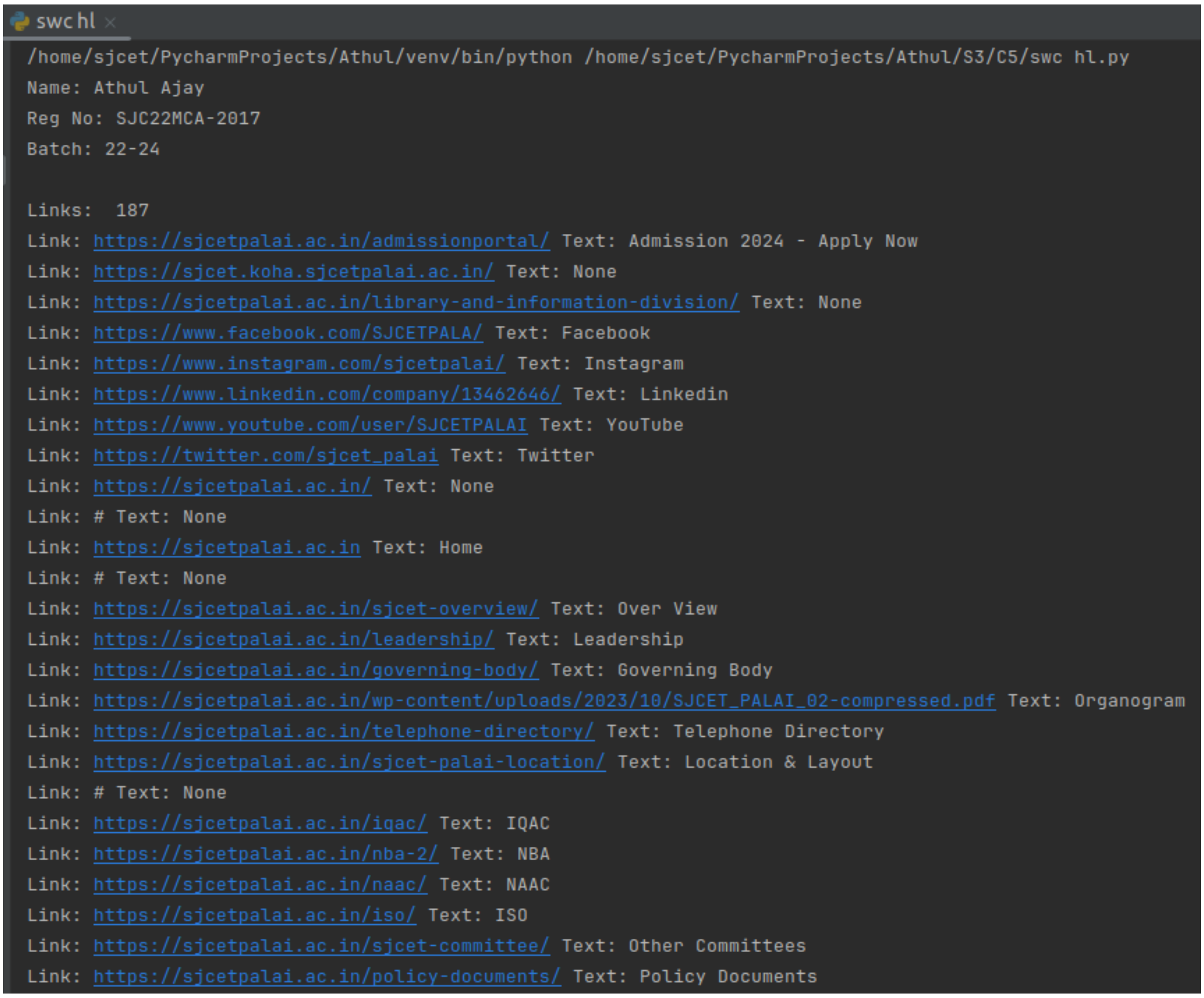
```
/home/sjcet/PycharmProjects/Athul/venv/bin/python /home/sjcet/PycharmProjects/Athul/S3/C5/swc p.py
Name: Athul Ajay
Reg No: SJC22MCA-2017
Batch: 22-24

P tag: 47

                W3Schools offers a wide range of services and products for beginners and professionals,

                helping millions of people everyday to learn and master new skills.

Enjoy our free tutorials like millions of other internet users since 1999
Explore our selection of references covering all popular coding languages

                      Create your own website with
                      W3Schools Spaces
                      - no setup required

Test your skills with different exercises
Test yourself with multiple choice questions
Document your knowledge

                      Create a
                      free
                      W3Schools Account to Improve Your Learning Experience

Track your learning progress at W3Schools and collect rewards
Become a PRO user and unlock powerful features (ad-free, hosting, videos,..)
Not sure where you want to start? Follow our guided path
With our online code editor, you can edit code and view the result in your browser
```

2. Write a program to implement simple web crawler using Python. Display all hyperlinks in the page

Input:

```
import requests
from bs4 import BeautifulSoup

def getdata(url):
    r = requests.get(url)
    return r.content

htmldata = getdata("https://sjcetpalai.ac.in/")
soup = BeautifulSoup(htmldata, 'html.parser')

print("Name: Athul Ajay")
print("Reg No: SJC22MCA-2017")
print("Batch: 22-24")
print()
links = soup.find_all("a")
print("Links: ", len(links))
for link in links:
    if link.get("href") != "":
        print("Link:", link.get("href"), "Text:", link.string
```

Output:



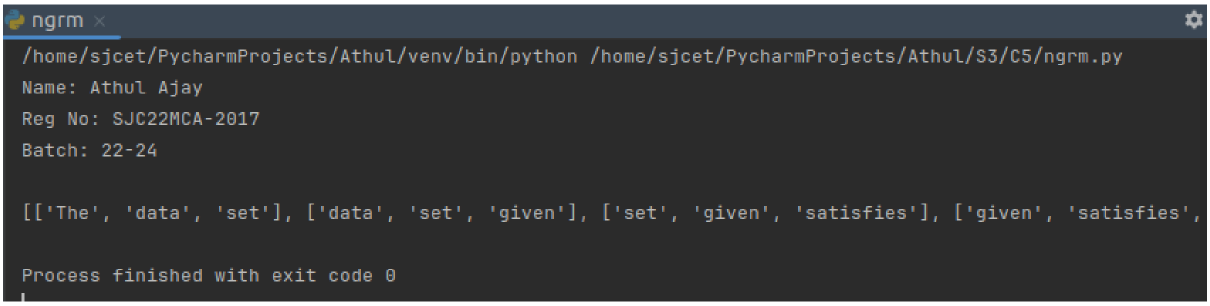
```
/home/sjcet/PycharmProjects/Athul/venv/bin/python /home/sjcet/PycharmProjects/Athul/S3/C5/swc hl.py
Name: Athul Ajay
Reg No: SJC22MCA-2017
Batch: 22-24

Links:  187
Link: https://sjcetpalai.ac.in/admissionportal/ Text: Admission 2024 - Apply Now
Link: https://sjcet.koha.sjcetpalai.ac.in/ Text: None
Link: https://sjcetpalai.ac.in/library-and-information-division/ Text: None
Link: https://www.facebook.com/SJCETPALA/ Text: Facebook
Link: https://www.instagram.com/sjcetpalai/ Text: Instagram
Link: https://www.linkedin.com/company/13462646/ Text: Linkedin
Link: https://www.youtube.com/user/SJCETPALAI Text: YouTube
Link: https://twitter.com/sjcet_palai Text: Twitter
Link: https://sjcetpalai.ac.in/ Text: None
Link: # Text: None
Link: https://sjcetpalai.ac.in Text: Home
Link: # Text: None
Link: https://sjcetpalai.ac.in/sjcet-overview/ Text: Over View
Link: https://sjcetpalai.ac.in/leadership/ Text: Leadership
Link: https://sjcetpalai.ac.in/governing-body/ Text: Governing Body
Link: https://sjcetpalai.ac.in/wp-content/uploads/2023/10/SJCET_PALAI_02-compressed.pdf Text: Organogram
Link: https://sjcetpalai.ac.in/telephone-directory/ Text: Telephone Directory
Link: https://sjcetpalai.ac.in/sjcet-palai-location/ Text: Location & Layout
Link: # Text: None
Link: https://sjcetpalai.ac.in/iqac/ Text: IQAC
Link: https://sjcetpalai.ac.in/nba-2/ Text: NBA
Link: https://sjcetpalai.ac.in/naac/ Text: NAAC
Link: https://sjcetpalai.ac.in/iso/ Text: ISO
Link: https://sjcetpalai.ac.in/sjcet-committee/ Text: Other Committees
Link: https://sjcetpalai.ac.in/policy-documents/ Text: Policy Documents
```

3. Program for Natural Language Processing which performs n-grams (without using library)

Input:

```
def gen_ngrams(text, WordsToCombine):
    words = text.split()
    output = []
    for i in range(len(words) - WordsToCombine + 1):
        output.append(words[i:i + WordsToCombine])
    return output
print("Name: Athul Ajay")
print("Reg No: SJC22MCA-2017")
print("Batch: 22-24")
print()
x = gen_ngrams(
    text= 'The data set given satisfies the requirement for model generation and s
used in Data Science Lab',
    WordsToCombine=3)
print(x)
```

Output:



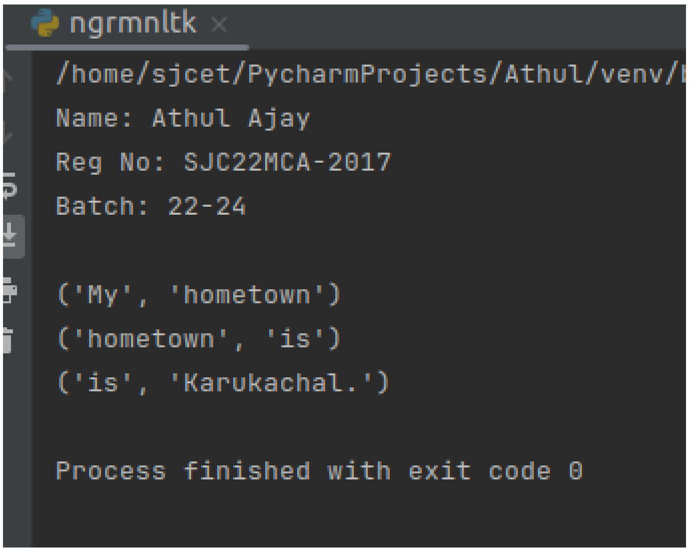
```
/home/sjcet/PycharmProjects/Athul/venv/bin/python /home/sjcet/PycharmProjects/Athul/S3/C5/ngrm.py
Name: Athul Ajay
Reg No: SJC22MCA-2017
Batch: 22-24

[['The', 'data', 'set'], ['data', 'set', 'given'], ['set', 'given', 'satisfies'], ['given', 'satisfies',

Process finished with exit code 0
```

4. Program for Natural Language Processing which performs n-grams (using nltk library)

Input:

```
print("Name: Athul Ajay")

print("Reg No: SJC22MCA-2017")

print("Batch: 22-24")

print()

from nltk import ngrams

sent = "My hometown is Karukachal."

n = 2

unigrams = ngrams(sent.split(), n)

for grams in unigrams:

    print(grams)
```

Output:



```
/home/sjcet/PycharmProjects/Athul/venv/
Name: Athul Ajay
Reg No: SJC22MCA-2017
Batch: 22-24

('My', 'hometown')
('hometown', 'is')
('is', 'Karukachal.')

Process finished with exit code 0
```

5. For given text,

 perform word

 sentence tokenization

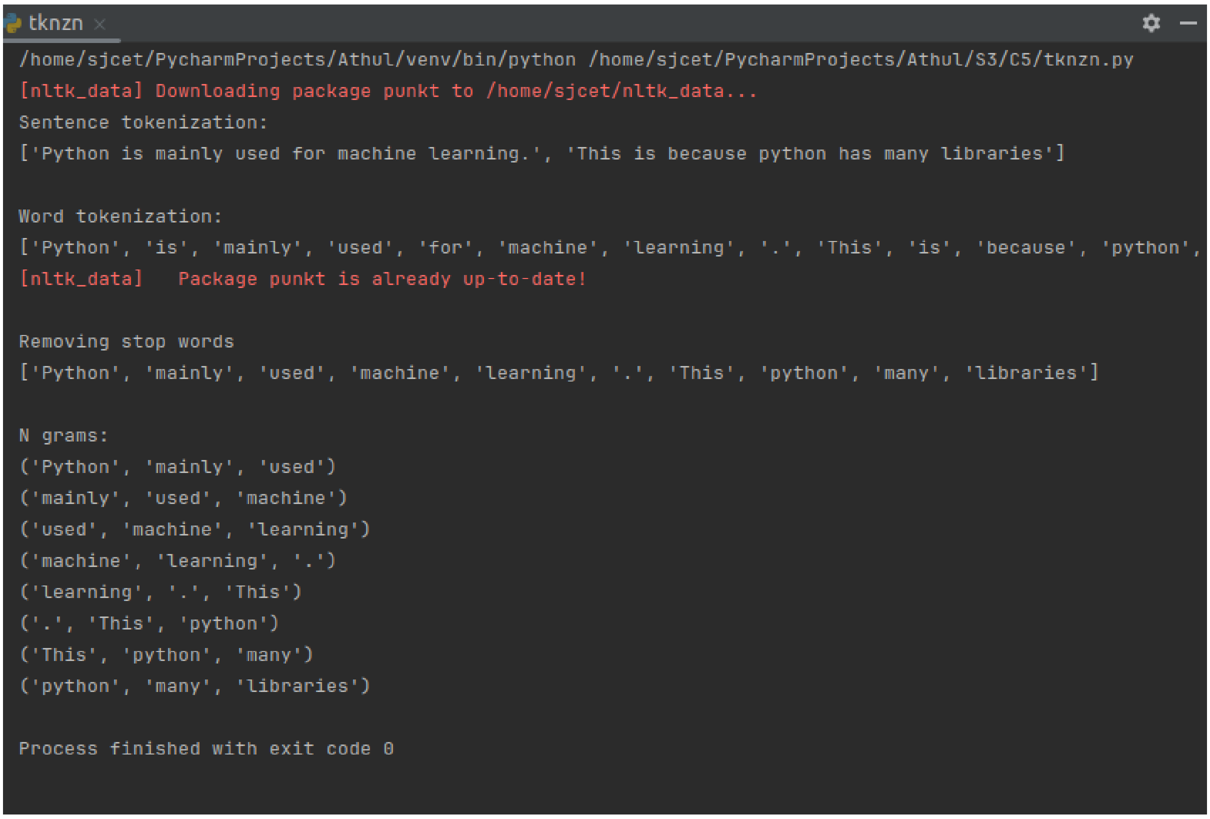
 Remove the stop words from the given text

 create n-grams

Input:

```
import nltk
from nltk import ngrams
from nltk.corpus import stopwords
from nltk.tokenize import sent_tokenize, word_tokenize
nltk.download('punkt')
txt1 = 'Python is mainly used for machine learning. This is because python has many libraries'
print('Sentence tokenization: ')
print(sent_tokenize(txt1))
print()
print('Word tokenization: ')
print(word_tokenize(txt1))
text = word_tokenize(txt1)
txt2 = [word for word in text if word not in stopwords.words('english')]
print()
print('Removing stop words')
print(txt2)
print()
print('N grams: ')
unigrams = ngrams(txt2, 3)
for grams in unigrams:
    print(grams)
```

Output:



```
/home/sjcet/PycharmProjects/Athul/venv/bin/python /home/sjcet/PycharmProjects/Athul/S3/C5/tknzn.py
[nltk_data] Downloading package punkt to /home/sjcet/nltk_data...
Sentence tokenization:
['Python is mainly used for machine learning.', 'This is because python has many libraries']

Word tokenization:
['Python', 'is', 'mainly', 'used', 'for', 'machine', 'learning', '.', 'This', 'is', 'because', 'python',
[nltk_data]   Package punkt is already up-to-date!

Removing stop words
['Python', 'mainly', 'used', 'machine', 'learning', '.', 'This', 'python', 'many', 'libraries']

N grams:
('Python', 'mainly', 'used')
('mainly', 'used', 'machine')
('used', 'machine', 'learning')
('machine', 'learning', '.')
('learning', '.', 'This')
('.', 'This', 'python')
('This', 'python', 'many')
('python', 'many', 'libraries')

Process finished with exit code 0
```

6. Given dataset contains 200 records and five columns, two of which describe the customer’s annual income and spending score. The latter is a value from 0 to 100. The higher the number, the more this customer has spent with the company in the past:

Using k means clustering create 6 clusters of customers based on their spending pattern.

□ Visualize the same in a scatter plot with each cluster in a different color scheme.

□ Display the cluster labels of each point (print cluster indexes)

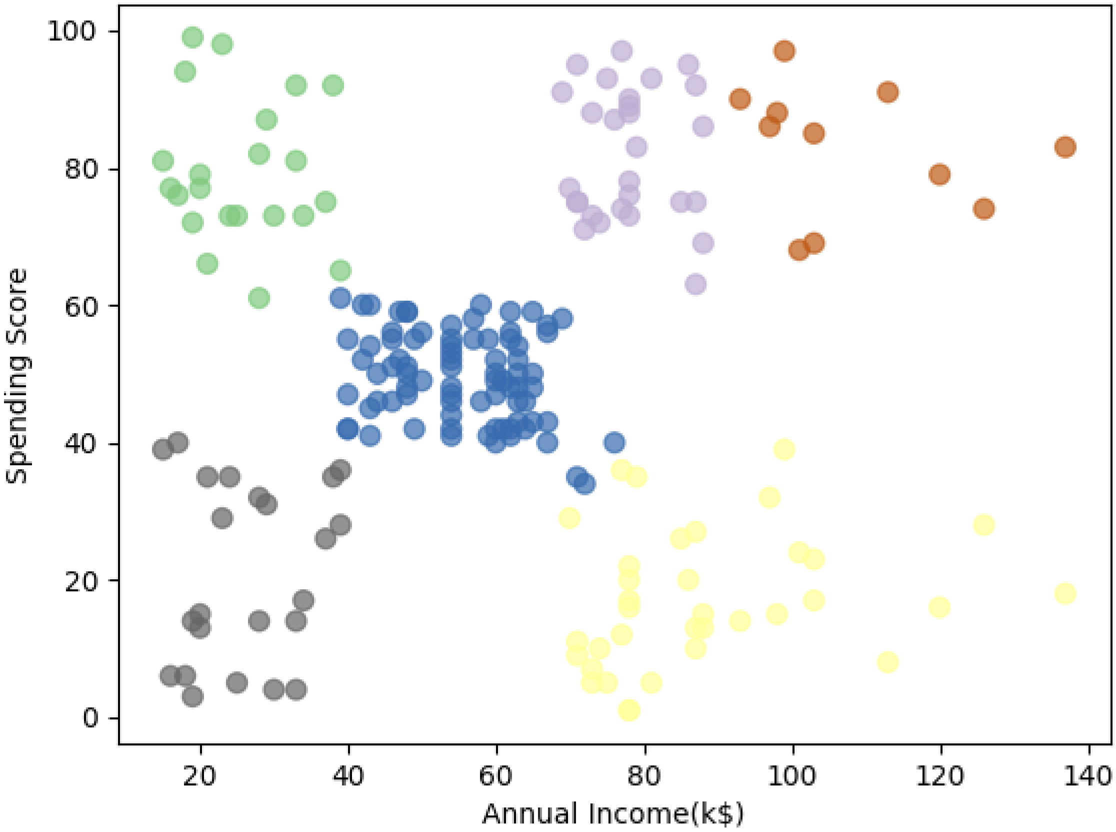
□ Display the cluster centers.

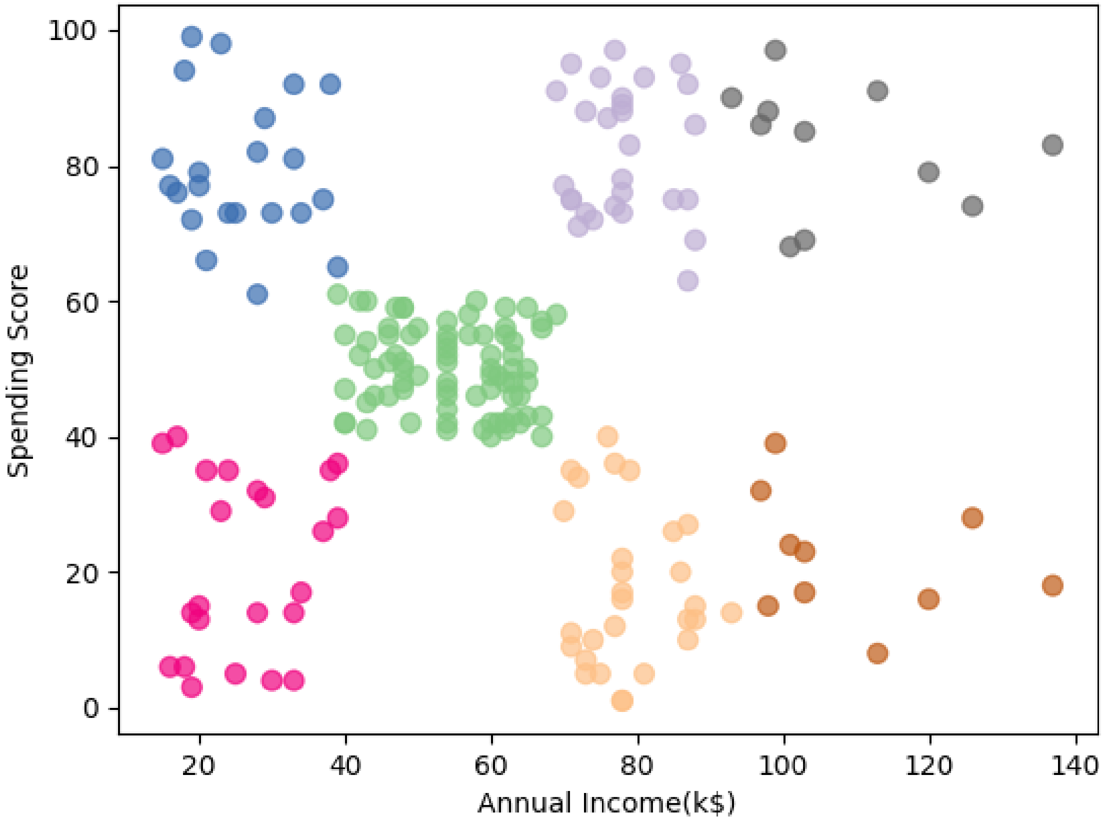
□ Use different values of K and visualize the same using scatter plot

Input:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import  KMeans
cust = pd.read_csv('customer_data.csv')
cust.head()
point = cust.iloc[:, 3:5].values
x = point[:, 0]
y = point[:, 1]
plt.scatter(x, y, s=50, alpha=0.7)
plt.xlabel('Annual Income(k$)')
plt.ylabel('Spending Score')
plt.show()

kmeans = KMeans(n_clusters=6, random_state=0)
kmeans.fit(point)
pred_clust_index =kmeans.predict(point)
plt.scatter(x, y, c=pred_clust_index, s=50, alpha=0.7, cmap='virdis')
plt.xlabel('Annual Income(k$)')
plt.ylabel('Spending Score')
plt.show()

center = kmeans.cluster_centers_
plt.scatter(center[:, 0], center[:, 1], c='red', s=100)
plt.xlabel('Annual Income(k$)')
plt.ylabel('Spending Score')
plt.show()

#displays 7 diff clusters
kmeans = KMeans(n_clusters=7, random_state=0)
kmeans.fit(point)
pred_clust_index =kmeans.predict(point)
plt.scatter(x, y, c=pred_clust_index, s=50, alpha=0.7, cmap='virdis')
plt.xlabel('Annual Income(k$)')
plt.ylabel('Spending Score')
plt.show()

center = kmeans.cluster_centers_
plt.scatter(center[:, 0], center[:, 1], c='red', s=100)
plt.xlabel('Annual Income(k$)')
plt.ylabel('Spending Score')
plt.show()
```

Output:


100
80
60
40
20
0
Spending Score
20
40
60
80
100
120
140
Annual Income(k$)


100
80
60
40
20
0
Spending Score
20
40
60
80
100
120
140
Annual Income(k$)